REMARKS

1. In response to the final Office Action mailed August 24, 2007, Applicants respectfully requests reconsideration. Claims 1, 3, 10, and 17 were previously canceled. Claims 2, 4-9, 11-16 and 18-22 were previously presented in the application. In the outstanding Office Action, all claims have been rejected. By the foregoing Amendments, independent claims 2, 9 and 16 have been amended. New claims 23-25 have been added. Thus, upon entry of this paper, claims 2, 4-9, 11-16, and 18-25 will be pending in this application. Based on the above Amendments and following Remarks, Applicant respectfully requests that all outstanding objections and rejections be reconsidered, and that they be withdrawn.

Art of Record

2. Applicants acknowledge receipt of form PTO-892 identifying additional references made of record by the Examiner.

Claim Rejections under 35 U.S.C. §103

3. Independent claims 2, 9 and 16; and dependent claims 4-8, 11-15, and 18-22 have been rejected under 35 U.S.C. § 103(a) as unpatentable over U.S. Patent No. 6,314,558 (Angel et al.) in view of "World Wide Web: Beyond the Basics" (Abrams et al.). Based upon the above Amendments and following Remarks, Applicants respectfully request reconsideration and withdrawal of these rejections.

4. Independent claims 2, 9 and 16 have been amended and new claims 23-25 have been added to clarify the invention. In particular, independent claim 2 has been amended to recite:

> communicating with a first interface to identify classes and selecting at least one method of said classes for instrumentation;
>
> installing hooks in the selected at least one method with instrumentation tools, said instrumentation tools reading and writing the classes in order to produce modified classes;
>
> inserting instrumentation code in a bytecode representation of the selected at least one method without modifying a source code of the selected at least one method, said hooks enabling said inserting of the instrumentation code;
>
> generating a wrapper method with said instrumentation tools that contains the instrumentation code and a call to the bytecode representation of the at least one method, wherein the instrumentation code comprises bytecodes;
>
> executing the bytecodes during execution of the at least

one method; and

generating a call, by the executed bytecodes, to a second interface wherein the call comprises information regarding the instrumented at least one method.

Claims 9 and 16 have been similarly amended. Support for the amendments and the new claims is provided by the original specification and figures. In particular, **FIG. 2** as shown below and the original specification disclose a system that includes a bytecode instrumentation engine that can be utilized to modify the bytecode associated with a Java application at any time prior to, or during, the loading and initialization of the bytecode by a Java virtual machine (**JVM**). More specifically, as disclosed by the specification, the bytecode instrumentation engine can include instrumentation tools, herein referred to as Bytecode Instrumentation Program (**BIP**) and the Bytecode Instrumentation Controller (**BIC**), that can modify methods of classes associated with a Java application prior to being loaded or as they are being loaded, respectively, by a **JVM**. As an example,

FIG. 2

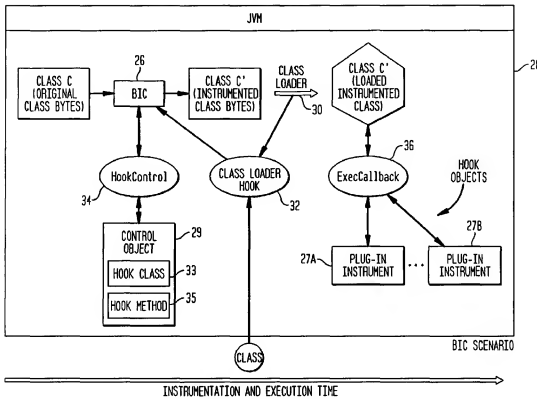

FIG. 2 schematically illustrates modification of an exemplary class C, and more specifically modification of one or more methods of class C, of a Java application by the **BIC** instrumentation tool **26** of the invention as the class is being loaded by a **JVM 28** running on an application server. The application server can employ a class loader **30** that provides a class loader hook control interface **32**. The class loader hook control interface **32** invokes the **BIC** instrumentation tool **26** to determine/identify whether any methods of the exemplary class C need to be instrumented. The **BIC** instrumentation tool **26** in turn determines whether any method(s) of class C is(are) slated for bytecode modification by communicating with a first interface **34** (i.e., a Hook Control interface) that directs the bytecode modification process. More particularly, the Hook Control interface **34** allows a user to identify selected classes or interfaces, and selected methods associated with these classes, for instrumentation;

and the ExecCallback interface **36** enables various types of monitoring tools to be plugged into a second interface and receive information when classes are executed.

5. In consideration of the discussion above, it is respectfully submitted that the amendments and the new claims are clearly supported by the original specification and figures and thus, raise no questions of new matter.

6. Angel et al. discloses code instrumentation is performed by adding statements to software (e.g., source code) in order to monitor performance and operation of the software during run time. In addition, Angel et al. discloses various systems compile source code into an interpretive language, such as byte code, and that though the *byte code is machine independent* so that it may be run on a computer that uses an interpreter to perform the operations indicated by the byte code, the interpreter is machine dependent.

7. In addition, Angel et al. discloses, as shown in **FIG. 12** below, a data flow diagram **400** that illustrates operation of a virtual machine (VM) runtime system that interprets and runs byte code, a class instantiator **402** may receive a class input and generates a class instance **406** and the class instance **406** is provided as an input to the VM runtime module **404** which interprets and executes the executable steps of the class instance **406**.

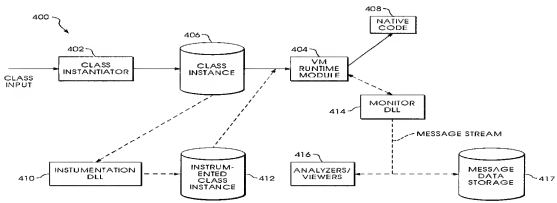

Fig. 12

8. Further, Angel et al. discloses a user can supplement the byte code provided in the class instance **406** with separate native code that may be used in conjunction with the byte code and that the byte code may be instrumented as the class is loaded by the VM runtime system. Furthermore, Angel et al. discloses the class instance **406** is provided to an instrumentation

DLL **410** which instruments the byte code of the class instance **406** to provide an instrumented class instance **412** and that the instrumented class instance **412** is provided as an input to the VM runtime module **404** instead of the class instance **406**.

9. Moreover, Angel et al. discloses, as shown in **FIG. 20** below, a flow chart **600** that illustrates instrumenting the class **406** of **FIG. 12** to provide the instrumented class **412**.

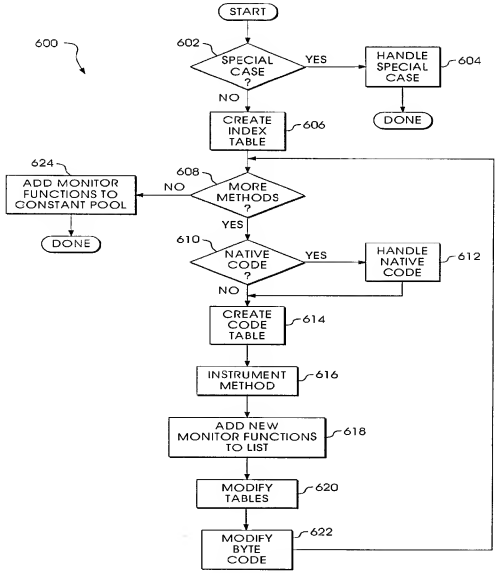

Fig. 20

10. In **FIG. 20**, processing begins at a test step **602** where it is determined if the class being instrumented. In particular, Angel et al. discloses control passes from the test step **608** to a test step **610** where it is determined if the method being processed is implemented in native

code and that different VM vendors provide different mechanisms for allowing native code to be called from byte code. Further, Angel et al. discloses, if it is determined at the test step **610** that the method is implemented in native code (by examining the access_flags in the class), control passes from the step **610** to a step **612** where instrumenting the native code is handled by, for example, adding a byte code wrapper to the method, wherein the wrapper causes the VM (and the instrumentation software) to treat the native code method as a conventional byte code method. Furthermore, Angel et al. discloses a processing at the step **612** may include modifying the native attribute of the method to convert the method to a byte code method, creating a new name for the native method, adding the new name as a private native method declaration, adding byte code instructions to call the native method under the new name.

11. However, Angel et al. nowhere discloses as amended claim 2 recites:

> *communicating with a first interface to identify classes and selecting at least one method of said classes for instrumentation;*
>
> *installing hooks in the selected at least one method with instrumentation tools, said instrumentation tools reading and writing the classes in order to produce modified classes;*
>
> *inserting instrumentation code in a bytecode representation of the selected at least one method without modifying a source code of the selected at least one method, said hooks enabling said inserting of the instrumentation code;*
>
> generating a wrapper method with said instrumentation tools that contain the instrumentation code and a call to the bytecode representation of the at least one method, wherein the instrumentation code comprises bytecodes;
>
> executing the bytecodes during execution of the at least one method; and
>
> *generating a call, by the executed bytecodes, to a second interface wherein the call comprises information regarding the instrumented at least one method* (emphasis added).

That is, Angel et al. does not disclose the italicized sections of claim 2, as recited above, and similarly worded sections of claims 9 and 16. Moreover, Angel et al. does not disclose the additional limitations of the "HookControl interface", "second interface," Bytecode Instrumentation Controller (BIC)" and "Bytecode Instrumentation Program (BIP)" of the new claims.

12. Thus, it is respectfully submitted for at least similar reasons to those discussed above with reference to independent claim 2, independent claims 9 and 16, and claims dependent thereon are not disclosed by Angel et al.

13. The outstanding Office Action acknowledges other deficiencies in Angel et al. and attempts to overcome those deficiencies by combining Abram et al. with Angel et al. In particular, the outstanding Office Action cites Abrams et al. as disclosing that Java is used to implement an operating system (OS), thereby making all native code accessible applications implemented as byte code within the Java OS." However, this disclosure by Abrams et al. alone cannot overcome all of the deficiencies of Angel et al., as discussed above. In particular, Abrams et al. nowhere discloses as amended claim 2 recites:

> *communicating with a first interface to identify classes and selecting at least one method of said classes for instrumentation;*
> *installing hooks in the selected at least one method with instrumentation tools, said instrumentation tools reading and writing the classes in order to produce modified classes;*
> *inserting instrumentation code in a bytecode representation of the selected at least one method without modifying a source code of the selected at least one method, said hooks enabling said inserting of the instrumentation code;*
> generating a wrapper method with said instrumentation tools that contain the instrumentation code and a call to the bytecode representation of the at least one method, wherein the instrumentation code comprises bytecodes;
> executing the bytecodes during execution of the at least one method; and
> *generating a call, by the executed bytecodes, to a second interface wherein the call comprises information regarding the instrumented at least one method* (emphasis added).

That is, Abrams et al. does not disclose the italicized sections of claim 2, as recited above, and similarly worded sections of claims 9 and 16. Moreover, Abrams et al. does not disclose the additional limitations of the "HookControl interface", "second interface," Bytecode Instrumentation Controller (BIC)" and "Bytecode Instrumentation Program (BIP)" of the new claims. Thus, Abrams et al. cannot overcome all of the deficiencies of Angel et al.

14. Therefore, it is respectfully submitted that neither Angel et al. nor Abrams et al., whether taken alone or in combination, disclose, suggest or make obvious the claimed

invention and that independent claims 2, 9 and 16, and claims dependent thereon, patentably distinguish thereover.

Dependent Claims

15. As discussed above, the dependent claims incorporate all of the subject matter of their respective independent claims and add additional subject matter which makes them *a fortiori* independently patentable over the art of record. Accordingly, Applicants respectfully request that the outstanding rejections of the dependent claims be reconsidered and withdrawn.

Conclusion

16. In view of the foregoing, this application should be in condition for allowance. A notice to this effect is respectfully requested.

Dated: November 21, 2007

Respectfully submitted,

Electronic signature: /Michael G. Verga/
Michael G. Verga
Registration No.: 39,410
CONNOLLY BOVE LODGE & HUTZ LLP
1875 Eye Street, N.W.
Suite 1100
Washington, DC 20006
(202) 331-7111 (Tel)
(202) 293-6229 (Fax)
Attorney for Applicant